PERC
performance evaluation research center
an integrated software infrastructure center

Concluding Remarks - Future Directions

Jack Dongarra
Innovative Computing Lab
University of Tennessee
http://www.cs.utk.edu/~dongarra/

SciDAC
Scientific Discovery through Advanced Computing

Software Technology & Performance

- **Tendency to focus on hardware**
- **Software required to bridge an ever widening gap**
- **Gaps between usable and deliverable performance is very steep**
  - **Performance only if the data and controls are setup just right**
    - **Otherwise, dramatic performance degradations, very unstable situation**
    - **Will become more unstable**
- **Challenge of Libraries, PSEs and Tools is formidable with Tflop/s level, even greater with Pflops, some might say insurmountable.**

The Need for Adaptivity

- Growing complexity of the systems we use threatens to undermine the benefits they aim to provide.
- We've relied mainly on human interactions to manage the complexity.
- With the complexity growing it is becoming beyond the ability to manage effectively.
- Hide the complexities while optimizing the resources.

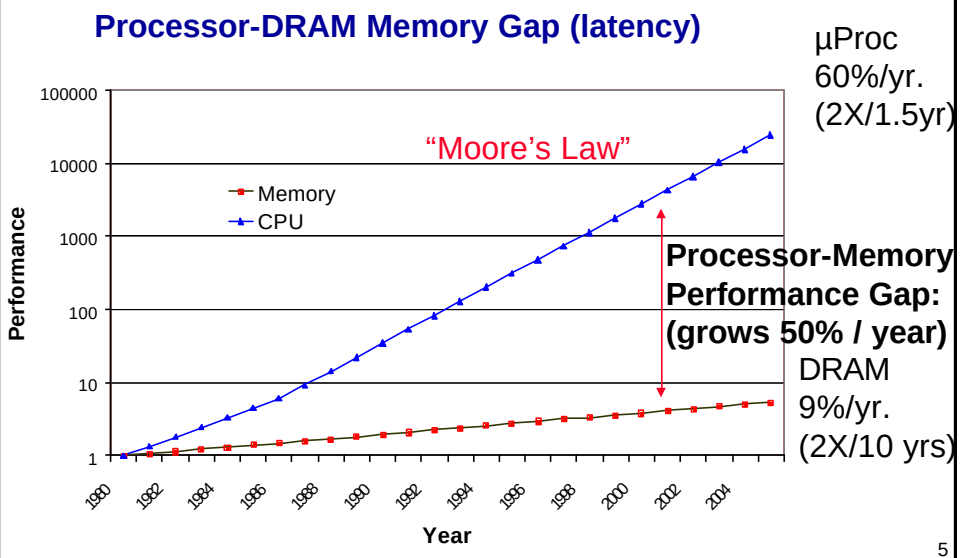
3

Types Of Adaptivity

- Adaptation to the environment
  - Processor: investigate processor hardware characteristics and optimize for them
  - Network: investigate connectivity, latency, bandwidth, congestion, load
- Adaptation to user data: investigate user data and make decisions based thereon
- Static adaptivity: adapt yourself to the environment during, potentially expensive, setup phase
- Dynamic adaptivity: at run-time adapt to current conditions, both user data and computational environment

4

Where Does the Performance Go? or Why Should I Care About the Memory Hierarchy?



5

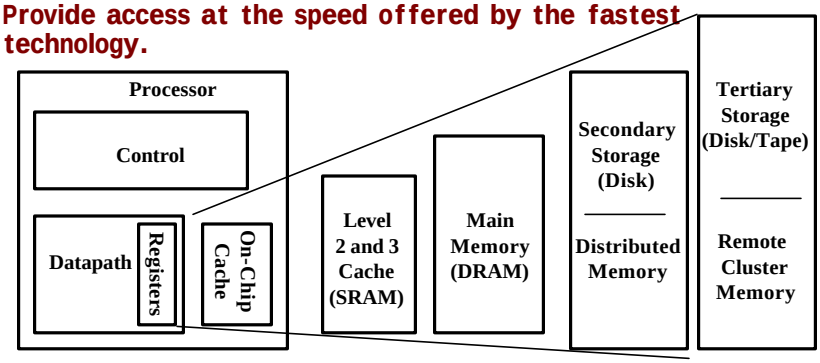
Optimizing Computation and Memory Use

- **Computational optimizations**
  - **Theoretical peak:(# fpus)*(flops/cycle) * Mhz**
    - **Pentium 4: (1 fpu)*(2 flops/cycle)*(2.53 Ghz) = 5060 MFLOP/s**
- **Operations like:**
  - $a = x^T y$: **2 operands (16 Bytes) needed for 2 flops; at 5060 Mflop/s will requires 5060 MWord/s bandwidth**
  - $y = a\,x + y$: **3 operands (24 Bytes) needed for 2 flops; at 5060 Mflop/s will requires 7590 MWord/s bandwidth**
- **Memory optimization**
  - **Theoretical peak: (bus width) * (bus speed)**
    - **Pentium 4: (32 bits)*(533 Mhz) = 2132 MB/s = 266 MWord/s**

6

Memory Hierarchy

- **By taking advantage of the principle of locality:**
  - **Present the user with as much memory as is available in the cheapest technology.**
  - **Provide access at the speed offered by the fastest technology.**



Speed (ns):	1s	10s	100s	10,000,000s (10s ms)	10,000,000,000s (10s sec)
Size (bytes):	100s	Ks	Ms	100,000 s (.1s ms)	10,000,000 s (10s ms)
				Gs	Ts

7

Self Adapting Software

- **Software system that …**
  - **Obtains information on the underlying system where they will run.**
  - **Adapts application to the presented data and the available resources perhaps provide automatic algorithm selection**
  - **During execution perform optimization and perhaps reconfigure based on newly available resources.**
  - **Allow the user to provide for faults and recover without additional users involvement**
- **The moral of the story**
  - **We know the concepts of how to improve things.**
  - **Capture insights/experience – do what humans do well**
  - **Automate the dull stuff**

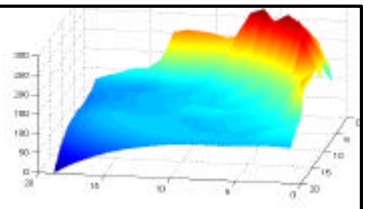
8

SANS
(Self Adapting Numerical Software)

- **Design a system that can adjust to varying circumstances and deal with the environment effectively.**
  - **Configure and perhaps reconfigure itself under varying and unpredictable conditions.**
  - **Optimize the operations to fit the environment.**
  - **Detect faults and recover gracefully.**

9

Software Generation Strategy - ATLAS BLAS

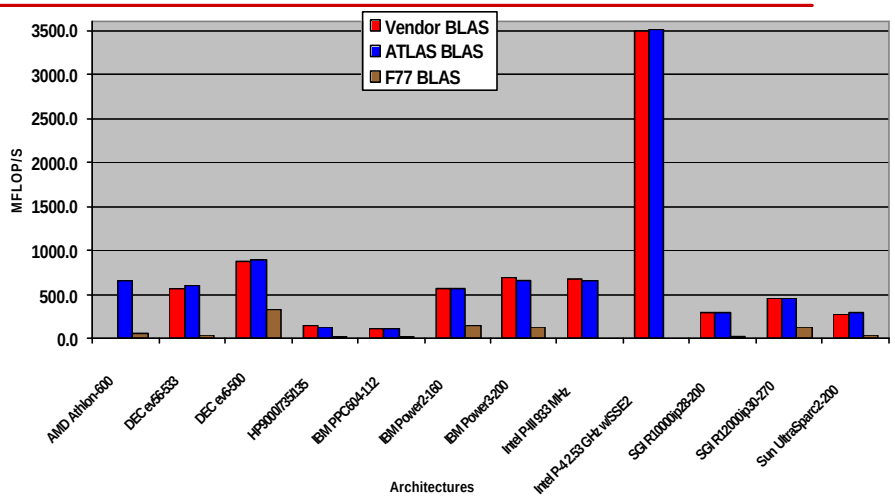


- **Parameter study of the hw**
- **Generate multiple versions of code, w/difference values of key performance parameters**
- **Run and measure the performance for various versions**
- **Pick best and generate library**
- **Level 1 cache multiply optimizes for:**
  - **TLB access**
  - **L1 cache reuse**
  - **FP unit usage**
  - **Memory fetch**
  - **Register reuse**
  - **Loop overhead minimization**

- **Takes ~ 20 minutes to run, generates Level 1,2, & 3 BLAS**
- **"New" model of high performance programming where critical code is machine generated using parameter optimization.**
- **Designed for modern architectures**
  - **Need reasonable C compiler**
- **Today ATLAS in used within various ASCI and SciDAC activities and by Matlab, Mathematica, Octave, Maple, Debian, Scyld Beowulf, SuSE,…**

10

ATLAS (DGEMM n = 500)

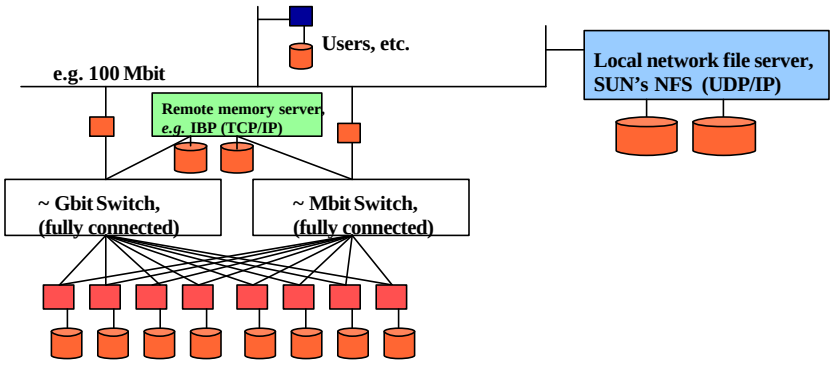


- ATLAS is faster than all other portable BLAS implementations and it is comparable with machine-specific libraries provided by the vendor.
- Looking at sparse operations

11

LAPACK For Clusters

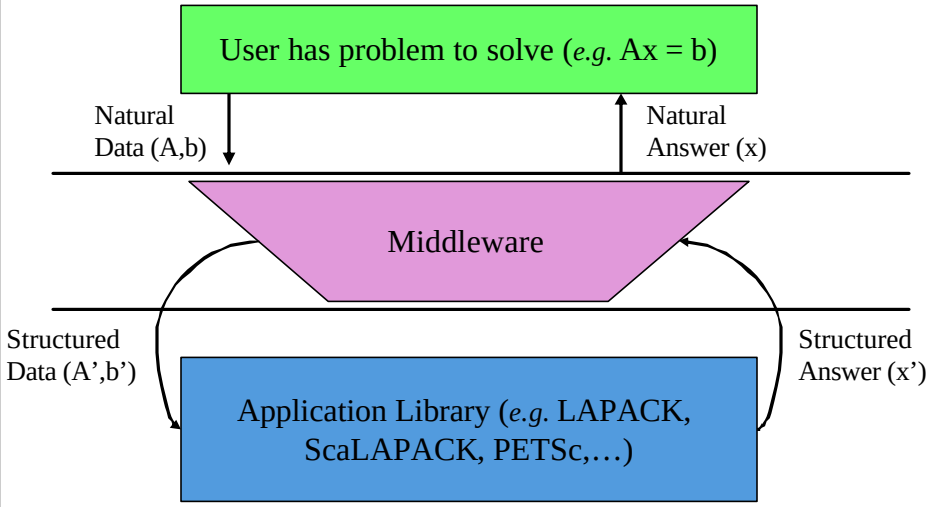
- Developing middleware which couples cluster system information with the specifics of a user problem to launch cluster based applications on the "best" set of resource available.



- Using ScaLAPACK as the prototype software, but developing a framework

12

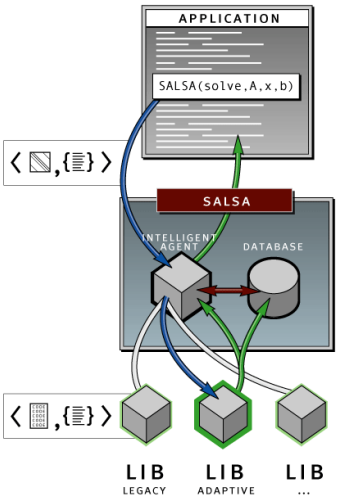
User Interface/Middleware



13

SALSA: Self-Adaptive Linear Solver Architecture

- Run-time adaptation to user data for linear system solving
- **Choice between direct/iterative solver**
  - **Space and runtime considerations**
  - **Numerical properties of system**
- **Choice of preconditioner, scaling, ordering, decomposition**
- **User steering of decision process**
- **Insertion of performance data in database**
- **Metadata on both numerical data and algorithms**
- **Heuristics-driven automated analysis**
- **Self-adaptivity: tuning of heuristics over time through experience gained from production runs**



Research Directions

- Parameterizable libraries
- Fault tolerant algorithms
- Annotated libraries
- Hierarchical algorithm libraries
- "Grid" (network) enabled strategies

A new division of labor between compiler writers, library writers, and algorithm developers and application developers will emerge.

15

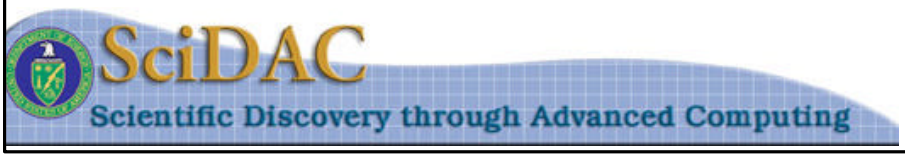
Future SANS Effort

- Intelligent Component
  - Automates method selection based on data, algorithm, and system attributes
- System component
  - Provides intelligent management of and access to clusters and computational grids
- History database
  - Records relevant info generated by the IC and maintains past performance data
- Fault Tolerant Aspect
  - Transparently detect and recover from failure
    - FT-MPI
    - Algorithmic Fault Tolerance

16

Questions?

- **Thanks for your participation**



17